Efficient and robust image descriptor for GUI object classification
Anastasia Dubrovina, Pavel Kisilev, Daniel Freedman, Sagi Schein, Ruth Bergman

HP Laboratories
HPL-2011-200

Keyword(s):
GUI object classification; image descriptor

Abstract:
Graphical User Interface (GUI) object classification is essential for image-based software automation tools. The challenges posed by GUI object classification are significantly different from those in natural image classification. In this paper we present a novel image descriptor developed specifically for GUI objects; it is robust to various changes in the appearance of GUI objects, such as various screen resolution, "skin", as well as various operating system related. We use this image descriptor with Support Vector Machine classifier, and experimentally show the descriptor robustness to the above transformations, and its superior performance compared to existing image descriptors.
Efficient and robust image descriptor for GUI object classification

Anastasia Dubrovina†, Pavel Kisilev‡, Daniel Freedman¶, and Sagi Schein, Ruth Bergman∥

Abstract. Graphical User Interface (GUI) object classification is essential for image-based software automation tools. The challenges posed by GUI object classification are significantly different from those in natural image classification. In this paper we present a novel image descriptor developed specifically for GUI objects; it is robust to various changes in the appearance of GUI objects, such as various screen resolution, "skin", as well as various operating system related. We use this image descriptor with Support Vector Machine classifier, and experimentally show the descriptor robustness to the above transformations, and its superior performance compared to existing image descriptors.

Keywords: GUI object classification, image descriptor.

1 Introduction

In this paper we address the problem of the Graphical User Interface (GUI) object classification. Examples of two types of GUI objects, namely push buttons and edit boxes, are presented in Fig. 1. Other examples of the GUI objects include radio buttons, check boxes, lists, menu item, etc. (see examples in Fig. 4). The GUI object classification is essential for software automation tools based on record/replay paradigm (such as [1]), where both the record and the replay stages are image-based, thus mimicking the human-computer interaction.

The main challenge in GUI object classification is the large variation in object’s appearance within a given class, even if the same operating system (OS) or browser are considered. For instance, the push buttons shown in Fig. 1 exhibit variety of sizes, colors, texts or images inside the objects. We assume that the GUI objects differ by geometric and "skin"-related transformations. The geometric transformations may be of two types: (1) scaling (potentially anisotropic) due to changes in screen resolutions, as well as naturally occurring differences such as edit box sizes, which can vary considerably in their aspect ratio; and (2) translation. We assume that the GUI objects in question were found (or segmented) using some segmentation algorithm, and that the object may not be centered within the subwindow provided by the segmentation. The "skin" changes are

† Technion, Israel Institute of Technology, nastyad@tx.technion.ac.il
‡ pavel.prvt@gmail.com
¶ Bar Ilan University, Israel, d.e.freedman@gmail.com
∥ HP Labs Israel, {sagi.schein, ruth.bergman}@hp.com
related to the variety of themes and colors provided by today’s operating systems, and these changes are somewhat harder to quantify. Our approach is to capture the look-and-feel information that remains relatively unchanged under "skin" transformations, and not color information that may change greatly.

In order to cope with the above challenges in GUI object classification task, we propose a novel type of image descriptor developed specifically for the GUI object classification task. The proposed descriptor is based on the 1D version of the Fourier-Mellin Transform. In addition, we incorporate information about object image gradients and the percentage of the white color (we explain the details in the next section). We show experimentally that the proposed descriptor is robust to various geometric and skin-related transformations of GUI objects. Using the Support Vector Machine (SVM) classifier [9] along with the proposed descriptor yields high correct classification rates for different GUI objects (see Section 3).

Note that in this paper we do not use the context of the GUI object, such as the surrounding objects, or the way the object changes its appearance in time, in order to perform the classification. Such context information can be successfully employed together with the proposed image descriptor to boost the classification accuracy, as proposed in [5].
1.1 Previous work

In the context of image-based software automation products, such as Project Sikuli [12], Prefab [4], EggPlant\(^1\), or related application for data masking - Magen\(^2\), the object recognition is based on some variant of simple image template matching, or text matching, and therefore, cannot cope with the variability of intra-class transformations we discussed above. Project Sikuli [12] also employs SIFT features [6], but these may not be informative for certain types of GUI objects, as we show in Section 3.

Many state-of-the-art object classification and recognition methods use standard features, such as SIFT, SURF [2], ASIFT [13], MSER [7], designed for natural images. Further, as opposed to other object classification frameworks, for instance, PASCAL VOC Challenge \(^2\), we deal with very small objects (size of a typical radio button is 12 × 12 pixels), of artificial nature. We have found that our image descriptor tends to work much better than the above features within the framework of relatively simple objects such as screen-shots and GUI objects.

The rest of the paper is organized as follows: we describe the details of the proposed image descriptor in Section 2. We provide the results of classification of different GUI objects, experimental evaluation of the descriptor robustness, and compare it to the SIFT descriptor in Section 3. We conclude the paper with Section 4.

2 Image descriptor

Schematic description of the proposed GUI object classifier is presented in Figure 2. For each GUI object we calculate the proposed image descriptor (described in details below), using the grayscale image of the object. In order to perform the classification, we first use a dataset of labeled GUI objects with their descriptors to train an SVM classifier, and then, given a new object, we calculate its descriptor and feed to the classifier to detect the object class.

In the following we describe the details of constructing the proposed image descriptor. It contains three different feature types.

1. Fourier-Mellin transform based feature. Given a grayscale image \(I\), we first calculate the derivatives of its projections on \(x\) and \(y\) axes, as follows

\[
I^x(x) = \frac{\partial}{\partial x} \left( \sum_y I(x, y) \right), \quad I^y(y) = \frac{\partial}{\partial y} \left( \sum_x I(x, y) \right).
\] (1)

\(^1\)http://www.testplant.com/products/eggplant/

\(^2\)http://pascalin.ecs.soton.ac.uk/challenges/VOC/
Then we apply a 1D Fourier–Mellin transform to it. Schematically, it can be represented as follows (shown here for $I^x$, but is the same for $I^y$):

$$I^x (x - x_0) \xrightarrow{\mathcal{F}} \mathcal{F}\{I^x\}(a \cdot u)e^{iu x_0} \xrightarrow{\mathcal{F}} |\mathcal{F}\{I^x\}(a \cdot u)| \xrightarrow{\log u} |\mathcal{F}\{I^x(\log u)\}| \xrightarrow{\mathcal{F}} |\mathcal{F}\{I^x(\log u)\}(v)|$$

First, we take the Fourier Transform of the 1D signal, that is of the derivatives of the image projections. Then we take its absolute value; this operation eliminates the horizontal and the vertical shifts in the spatial domain. We then use only half of the spectrum, that is the positive frequencies, as it is symmetric anyway. Next, we take the log scale in the frequency domain; this operation converts the scale into a new shift. Similarly, taking the second Fourier Transform and the absolute value eliminates the scale dependence. (Again, we use only positive frequencies). At the end, we get a shift and scale invariant representation, and it’s length is $\frac{1}{4}$ of the length of the original signal (since we halved the length by using twice only positive frequencies). In order to obtain a fixed-length representation for image of arbitrary size, we zero-pad $I^x$ to have 1024 samples (or downsample it to have 1024 samples, if $I^x$ is larger) before we calculate the first Fourier transform. Thus the result of this 1D Fourier-Mellin transform is a vector of 256 samples. After we concatenate transformed $I^x$ and $I^y$ signals, we have a 512-dimensional descriptor.

2. **Histogram of gradient directions (angles) based feature** We calculate the angles of gradients in the grayscale image, wrap them to be in the range $[0, \pi)$, and calculate their 4 bin histogram. We weight the gradient angles by their magnitude, and also multiply them by triangle window weights.
Efficient and robust image descriptor for GUI object classification

3. Percentage of white pixels

This descriptor calculates the percentage of the pixels with gray values larger than a predefined threshold (we used 245 for this). This descriptor helps to separate edit boxes and push buttons or other rectangular objects which are not empty (as edit boxes are). (In general, the above threshold can be learn from the database, though we did not do it).

Concatenating the above three types of features results in a 518-dimensional vector descriptor. Figure 3 illustrates the process of descriptor calculation. Note that the proposed descriptor size is independent of the original object image size.

Finally, we would like to use the above image descriptor to perform the classification of GUI objects using the SVM classifier. We use use the non-linear version of SVM, namely Kernel SVM, where the kernel is taken to be the commonly used Gaussian kernel between the descriptor vectors. That is, given two images of GUI objects, $I_1$ and $I_2$, and their descriptors $f_1$ and $f_2$, respectively, we measure the similarity between $I_1$ and $I_2$ by

$$K(f_1, f_2) = \exp(-\gamma \|f_1 - f_2\|_2^2), \gamma > 0. \quad (3)$$

The Gaussian bandwidth $\gamma$ is chosen by a cross-validation procedure.

3 Experimental results

In this section we show the results of the GUI object classification obtained with the proposed image descriptor. To perform the descriptor evaluation we
created a dataset of 258 segmented objects belonging to five GUI object classes:
check box, edit box, list box opener, push button and radio button. Examples of
objects from these five classes are shown in Fig. 4. We used this dataset both to
train the Kernel SVM classifier (details in the previous Section), and to perform
the cross-validation procedure to find the optimal SVM parameters, as suggested
in [3]. In our code, we used the LibSVM Kernel SVM implementation [3].

3.1 Classification with the proposed image descriptor

In order to evaluate quality of the classification with the proposed descriptor we
randomly divided the above dataset into non-overlapping training and testing
sets. We used training sets of different sizes - 40% – 90% of the whole dataset
size. We performed 10 experiments for each one of these training set sizes - each
time we randomly drew the necessary number of the training examples from
the dataset. We then calculated the average number of the correctly classified
objects from all five classes, normalized by the test set size, as a function of
the the training set size. The results are shown in Fig. 5, by the solid line with
diamond-shaped markers.

3.2 Descriptor persistence

In order to evaluate the robustness of the proposed descriptor to geometric
transformations we assembled a set of shifted, scaled and incorrectly segmented
versions of an object from our dataset (see Fig. 6). We then trained the SVM classifier using the objects from our dataset, excluding the transformed push buttons, and classified transformed version of our object. All of them were correctly classified as push buttons. Note that since we are interested in GUI objects, their segmentation can be made relatively accurately, therefore examples of incorrect segmentation in Fig. 6) may not happen in practice. Nevertheless, we saw that we can perform their classification even in those extreme cases as well.

### 3.3 Comparison with SIFT descriptors

In this section we compare the proposed descriptor to the state-of-the-art SIFT descriptor [6]. In order to perform the classification we adopted the Bag Of (Visual) Words (BOVW) approach used in [10]. As the GUI objects in question are very small, we cannot rely on SIFT feature detector. Also, the high inter-class variability in object appearance causes the SIFT feature detector find inconsistent features along the same class (say, text in push button - which is totally not representative Fig. 7 shows an example of several push buttons with SIFT features calculated using Vedaldi’s implementation of SIFT algorithm [11] for Matlab®, with grayscale object images. No features at all were found for some of the smaller buttons.
Therefore, instead of searching for SIFT features, we used the image pixels with high gradient magnitude value (higher than a predefined threshold of 10) as feature points. For these feature points we calculated modified SIFT descriptors: we used fixed windows of size $8 \times 8$ around each feature point, and calculated gradient histograms in its four subwindows of size $4 \times 4$, thus obtaining descriptors of length 16. We then performed vector quantization of collection of the descriptors obtained for all the objects, into 500 classes, and used the quantized vectors as visual words. Finally, the object descriptors were calculated as histograms calculating the number of appearances of the above visual words in each one of the object images.

We performed the classification as described in Section 3.1, using the above image descriptors. The graph of the correct classification percentage is shown in Fig. 5 above, dashed line, together with the correct classification percentage obtained using the proposed descriptors. Though the BOVW with SIFT descriptors performs quite well, the proposed descriptor yields better classification.

4 Conclusion

We presented a novel image descriptor designed for GUI object classification. We showed how it can be coupled with the Support Vector Machine classifier, to achieve robust classification of GUI objects from several selected GUI object classes, comparable to this of the existing image descriptors.
References


